From Javier

Packaging and ease of use

Currently it is not easy to :

* use europa for the first time
* debug models (look at constraint violations and search behavior)
* embed a europa planner into other applications, specially ones with UIs
* quickly prototype a planning/scheduling/constraint satisfaction application that uses europa

This can be addressed by :
* Finishing EUROPA documentation : User's Guide, Installation Guide and Tutorial
* Exposing a smaller interface for clients so that embedding is easier and UI tools can be built. Interface needs to be exposed in languages with strong UI/app building support like Java and/or Python. The DSA interface with some extensions can play this role, implementation is already in progress.
* Providing UI developer tools and UI components that can be quickly assembled for demos/applications. The action item is to build UI components on top of the DSA interface and make them available through planworks for developers and compatible with application frameworks like Ensemble and MCT for end-user applications
* Giving access to constraint violation information. The work done in DynamicEuropa for passive detection should be incorporated as part of the core, also eliminating the need for modeling changes. Violation information should be easily accessible through the DSA interface, a user should be able to say something as simple as "activity.getViolations()" to get constraint violations, this is currently very hard, but it is critical for debugging models and for building end-user applications.
* Longer term : binary packaging and interpretation instead of code generation should make it even easier for people to adopt, embed, debug and develop with europa

ANML

The ANML specification is now available, we can move forward with implementation. The main motivation for ANML :

* Provide support for the incorporation of reachability heuristics
* Provide a unified modeling platform for EUROPA and ASPEN
* ANML is intended to be a "better" (richer, more natural, concise while preserving advantages of existing languages) modeling language than either NDDL or AML. It will also be easier for people used to PDDL to understand it, so that the interaction with the planning research community can be more vigorous

Better built-in search support

Currently, there is very little built-in support for search. There are at least a couple of aspects that are problematic with this :
* Getting search to work is the bulk of the work for most applications. There are many general-purpose search techniques that can and should be reused, every europa user should not start from scratch solving every new problem.
* If the core europa team doesn't solve relevant search problems, how do we know that europa can do it? Even in the case when europa can do it, it may require such a detailed knowledge of europa that it effectively puts it out of the reach of most users. Finally, solving search problems will provide valuable insight into what extensions/packing

is needed in this area

This can be addressed by:

* Packaging of reachability heuristics enabled by ANML
* Packaging support for Local Search, a pre-requisite for this is native violation information support mentioned in the first section above
* Packaging solvers for relevant scheduling problems like RCPSP.
* Moving min-perturbation heuristic work done by DynamicEuropa team to core
* Solution to Becnhmark problems as part of the europa distribution. we should pick representative ones here. RCPSP for scheduling, probably some from the planning competitions
* Longer term : support for search modeling in the laguages (NDDL/ANML). framework for inter-leaving planning and scheduling search approaches.

EUROPA as FLIGHT Software

See Jeremy's notes on this. This looks like a major effort so I expect it would be staffed and scoped in its own separate thread, but of course we'll need to figure out how to make the current implementation as flight-friendly as possible once the effort gets underway.